

Confirmation No. 7347

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	SHRIVASTAVA, <i>et al.</i>	Examiner:	Knoll, C.
Serial No.:	10/566,515	Group Art Unit:	2111
Filed:	January 30, 2006	Docket No.:	US030254US2
Title:	MICROCONTROLLER WITH AN INTERRUPT STRUCTURE HAVING PROGRAMMABLE PRIORITY LEVELS WITH EACH PRIORITY LEVEL ASSOCIATED WITH A DIFFERENT REGISTER SET		

APPEAL BRIEF

Mail Stop Appeal Brief-Patents
Commissioner For Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Customer No. 65913

Dear Sir:

This Appeal Brief is submitted pursuant to 37 C.F.R. §41.37, in support of the Notice of Appeal filed February 29, 2008 and in response to the rejections of claims 1-21 as set forth in the Final Office Action dated November 29, 2007, and in acknowledgment of the Advisory Action dated February 27, 2008.

Please charge Deposit Account number 50-0996 (NXPS.281PA) \$510.00 for filing this brief in support of an appeal as set forth in 37 C.F.R. §1.17(c). If necessary, authority is given to charge/credit Deposit Account 50-0996 additional fees/overages in support of this filing.

I. Real Party In Interest

The real party in interest is NXP Semiconductors. The application is presently assigned of record, at reel/frame nos. 017529/0075 to Koninklijke Philips Electronics, N.V., headquartered in Eindhoven, the Netherlands. We have been authorized by both the assignee of record and NXP Semiconductors to convey herein that the entire right, title and interest of the instant patent application has been transferred to NXP Semiconductors.

II. Related Appeals and Interferences

While Appellant is aware of other pending applications owned by the above-identified Assignee, Appellant is unaware of any related appeals, interferences or judicial proceedings that would have a bearing on the Board's decision in the instant appeal.

III. Status of Claims

Claims 1-21 stand rejected and are presented for appeal. A complete listing of the claims under appeal is provided in an Appendix to this Brief.

IV. Status of Amendments

No amendments have been filed subsequent to the Office Action dated November 29, 2007.

V. Summary of Claimed Subject Matter

Appellant's recited invention relates to systems and methods, and to storage media for use therein, that implement an interrupt structure having programmed priority levels, each priority level being associated with a different register set.

Commensurate with independent claim 1, an example embodiment of the present invention is directed to a system that includes a processor (*see, e.g.*, processor 180 shown in Fig. 1, along with paragraph 0021 on page 4:8-17) that executes logical or arithmetic operations (*see, e.g.*, paragraph 0003 on page 1:10-24), a plurality of register bank blocks (*see, e.g.*, register bank blocks 120, 121, 122, 123, 124 and 125 shown in Fig. 2, along with paragraph 0025 to 0028 on page 7:20 to page 8:18) used as special function registers by the

processor during the execution of the logical or arithmetic operations (*see, e.g.*, special function register block 501 shown in Fig. 5, along with paragraph 0030 on page 8:28 to page 9:3), and a register bank block decoder circuit (*see, e.g.*, decoder circuit 140 shown in Fig. 2 and 701 shown in Fig. 7, along with paragraph 0033 on page 11:3-14) for activating one and only one of the plurality of register bank blocks (*see, e.g.*, paragraph 0006 on page 2:13-18), the register bank block decoder circuit responsive to interrupt event operations for selecting the one of the plurality of register bank blocks for being activated (*see, e.g.*, paragraph 0006 on page 2:13-18 and paragraph 0033 on page 11:3-14), where different interrupt event operations result in selection of different ones of the plurality of register bank blocks (*see, e.g.*, paragraph 0006 on page 2:13-18 and paragraph 0033 on page 11:3-14).

Commensurate with independent claim 13, an example embodiment of the present invention is directed to a method of switching processing resources in a data processing system, which includes the steps of providing a plurality of register bank blocks (*see, e.g.*, register bank blocks 120, 121, 122, 123, 124 and 125 shown in Fig. 2, along with paragraph 0025 to 0028 on page 7:20 to page 8:18), utilizing a first register bank block from the plurality of register bank blocks as special function registers during execution of logical or arithmetic operations (*see, e.g.*, special function register block 501 shown in Fig. 5, along with paragraph 0030 on page 8:28 to page 9:3), receiving of an interrupt request for initiating an interrupt event (*see, e.g.*, paragraph 0007 on page 2:19-27 and paragraph 0033 on page 11:3-14), determining if the interrupt request is to be fulfilled (*see, e.g.*, paragraph 0007 on page 2:19-27), and if so, selecting a second register bank block from the plurality of register bank blocks (*see, e.g.*, paragraph 0007 on page 2:19-27 and paragraph 0033 on page 11:3-14), the selected second register bank block in isolation from the first register bank block (*see, e.g.*, paragraph 0007 on page 2:19-27), and utilizing the second register bank block from the plurality of register bank blocks as special function registers during execution of logical or arithmetic operations (*see, e.g.*, paragraph 0007 on page 2:19-27 and paragraph 0033 on page 11:3-14).

Commensurate with independent claim 13, an example embodiment of the present invention is directed to a storage medium (*see, e.g.*, paragraph 0008 on page 3:1-8) having data stored thereon, the data for implementation of a processing system that includes first

instruction data (*see, e.g.*, paragraph 0008 on page 3:1-8 and paragraph 0025 on page 7:20-25) for providing a plurality of register bank blocks that are used as special function registers during execution of logical or arithmetic operations (*see, e.g.*, special function register block 501 shown in Fig. 5, along with paragraph 0030 on page 8:28 to page 9:3), and second instruction data (*see, e.g.*, paragraph 0008 on page 3:1-8 and paragraph 0026 on page 7:30-35) for providing a register bank block decoder circuit for activating one of the plurality of register bank blocks in isolation (*see, e.g.*, paragraph 0008 on page 3:1-8), the register bank block decoder circuit responsive to interrupt event operations for selecting the one of the plurality of register bank blocks for being activated (*see, e.g.*, paragraph 0008 on page 3:1-8 and paragraph 0033 on page 11:3-14), where different interrupt event operations result in selection of different ones of the plurality of register bank blocks (*see, e.g.*, paragraph 0008 on page 3:1-8 and paragraph 0033 on page 11:3-14).

As required by 37 C.F.R. § 41.37(c)(1)(v), a concise explanation of the subject matter defined in the independent claims involved in the appeal is provided herein. Appellant notes that representative subject matter is identified for these claims; however, the abundance of supporting subject matter in the application prohibits identifying all textual and diagrammatic references to each claimed recitation. Appellant thus submits that other application subject matter, which supports the claims but is not specifically identified above, may be found elsewhere in the application. Appellant further notes that this summary does not provide an exhaustive or exclusive view of the present subject matter, and Appellant refers to the appended claims and their legal equivalents for a complete statement of the invention.

VI. Grounds of Rejection to be Reviewed Upon Appeal

- A. Claims 1-7, 11, 13-16 and 19-21 stand rejected under 35 U.S.C. § 103(a) over Mitsuhiro (U.S. Patent No. 5,155,583) in view of Yoshida (U.S. Patent No. 5,450,566).
- B. Claims 8-10 and 17-18 stand rejected under 35 U.S.C. § 103(a) over Mitsuhiro (U.S. Patent No. 5,155,583) in view of standard register use, and in further view of Fujimura (U.S. Patent No. 5,751,988).
- C. Claim 12 stands rejected under 35 U.S.C. § 103(a) over Mitsuhiro (U.S. Patent No. 5,155,583) in view of standard register use, and in further view of Hohl (U.S. Patent No. 6,035,422).

VII. Argument

As set forth below, Appellant submits that the claimed invention is allowable over the cited references because the obviousness rejections are based on art that teaches away from, and that fails to provide correspondence to, the claimed invention. In particular, the cited references do not disclose the use of special function registers, and the use of special function registers is contrary to the teachings of the primary Mitsuhiro reference used in each of the rejections.

A purported obviousness rejection based on a combination of references fails unless the references are properly combinable and teach or suggest all the recited claim elements. Without a reasonable expectation of success and a valid reason for combining, references are not properly combinable. These conditions cannot be satisfied when a reference teaches away from the proposed combination or modification, or a reference is rendered inoperable for its intended purpose upon making the proposed combination or modification. Appellant submits that the Examiner's obviousness rejections fail to meet the required criteria.

Appellant further submits that the Examiner's rejections are predicated on conjecture and opinion without proper analysis of correspondence between the cited art and the features of the claims, without properly articulating how the proposed combinations would be made, and without fully addressing the substance of Appellant's arguments. Appellant therefore requests that the Board reverse the rejections.

A. The § 103(A) Rejection Of Claims 1-7, 11, 13-16 And 19-21 Over Mitsuhiro In View Of Yoshida Is Improper And Should Be Reversed.

1. The proposed combination does not correspond to all the features of the claims.

Appellant respectfully submits that the cited portions of the Mitsuhiro reference, taken alone or in view of Yoshida, do not correspond to the claimed invention. For example, the Mitsuhiro reference does not teach or suggest the use of special function registers in the manner recited in the claimed invention. Mitsuhiro discloses a data memory (36) that includes banks of general registers (*see, e.g.*, Mitsuhiro's Figure 1; Col. 4:15-21). Nowhere does Mitsuhiro disclose or suggest the use of special function registers in place of the general registers. Moreover, Yoshida includes no disclosure of special function registers, and thus does not cure the deficiency of Mitsuhiro (Appellant notes that Yoshida is cited only for allegedly disclosing performance of arithmetic operations on register data, and was not asserted in the Final Office Action for disclosing special function registers).

As Appellant demonstrated in the response of September 7, 2007, special function registers, as known to those of skill in the art and as consistent with Appellant's Specification, are accessed by a processor as if they were internal memory. *See, e.g.*, U.S. Patent No. 5,734,857 in the Summary of the Invention section, and the printout from http://www.hobbyprojects.com/8051_tutorial/special_function_registers.html (which was attached in Appellant's September 7, 2007 response, and is re-attached hereto for convenience). In contrast, the memory 36 disclosed by Mitsuhiro is used to temporarily store a copy of data from program status word (PSW) 20 and program counter 18. The temporarily stored data is then restored to PSW 20 and PC 18 rather than accessed as internal memory. As such, Mitsuhiro's description of the use of the registers within memory 36 is consistent with the term "general register" as used by Mitsuhiro, and is inconsistent with the term "special function register" as used by Appellant and as understood in the art.

An object of certain embodiments of Appellant's invention is to facilitate the execution of an interrupting program stream without storing and restoring interrupted program stream critical data (*see, e.g.*, Appellant's Specification, Paragraph 0005), whereas the cited portions of Mitsuhiro teach storing and restoring critical data in response to interrupt requests. The cited portions of the Mitsuhiro reference teach that, in response to an interrupt signal, CPU 16 operates to save the contents of PC 18 and PSW 20 in the selected register bank (*i.e.*, register banks 1, 2 and 3 of data memory 36). *See, e.g.*, Mitsuhiro Figures 1 and 2; Col. 6:3-12. The register banks taught by Mitsuhiro are used to store data while interrupt requests are processed by the CPU 16. Mitsuhiro's register banks are not used as special function registers by CPU 16 during the execution of logical or arithmetic operations as recited in the claimed invention.

In the Advisory Action, the Examiner responds to Appellant's arguments that the proposed combination fails to teach special function registers by stating that Yoshida uses registers to perform what the Examiner deems to be "special functions." The passage of Yoshida cited for support (*i.e.*, Yoshida Col. 4:45-48) merely discloses an instruction to add the contents of two registers and store the result in a third register. The Examiner makes the conclusory statement that such disclosure adequately sets forth special function registers, without providing any documentary evidence to support such an opinion. In particular, the Examiner has not addressed Appellant's explanation of special function registers as set forth above and in previous responses, much less provided any basis for how the registers disclosed in either Mitsuhiro or Yoshida might conform thereto. Appellant further submits that the Examiner's failure to provide a full explanation is contrary to M.P.E.P. § 707.07(f) which states, "[i]n order to provide a complete application file history and to enhance the clarity of the prosecution history record, an examiner must provide clear explanations of all actions taken by the examiner during prosecution of an application."

For these reasons, the proposed combination of Mitsuhiro in view of Yoshida cannot be properly read as disclosing the use of special function registers as claimed by Appellant. Therefore, Appellant submits that the § 103(a) rejection is improper and should be reversed.

2. No valid reason has been presented for modifying Mitsuhiro in the proposed manner.

Appellant respectfully submits that no valid reason has been presented for combining Mitsuhiro with Yoshida in a manner that would result in Appellant's invention. The Examiner admits that Mitsuhiro fails to disclose the use of registers for special functions, but argues that one of skill in the art would modify Mitsuhiro in view of Yoshida's alleged disclosure of registers that have particular functions during the execution of arithmetic operations. The only rationale provided by the Examiner in support of modifying Mitsuhiro in such a manner is that it "allows for enhancements, such as efficient use of register addressing modes." *See* Final Office Action, page 2.

As an initial matter, Appellant has been unable to ascertain what the Examiner's alleged modification of Mitsuhiro involves, and has requested clarification. For instance, to the extent that Yoshida discloses register data is being used in arithmetic operations, the Examiner has not provided information explaining what registers are being referenced, where the arithmetic operations are to be implemented, or how efficient use of register addressing modes would be seen as a result of any combination.

Appellant further submits that the § 103(a) rejection fails because the Examiner's asserted modification undermines the operation of the Mitsuhiro reference. As indicated in M.P.E.P. § 2143.01, when the asserted modification would undermine both the operation and the purpose of the main reference, the §103 rejection is improper. *See also In re Gordon*, 733 F.2d 900, 221 U.S.P.Q. 1125 (Fed. Cir. 1984) (A §103 rejection cannot be maintained when the asserted modification undermines the operation and/or purpose of main reference.). As Appellant noted above, Mitsuhiro discloses that the register banks are used in a manner that is inconsistent with the use of special function registers. As such, to the extent that the Examiner views Yoshida's register functions as "special," any modification to convert Mitsuhiro's registers into special function registers would alter Mitsuhiro's disclosed mode of operation.

Therefore, Appellant submits that no valid reason to modify the Mitsuhiro reference using the teachings of Yoshida has been presented. Therefore, Appellant submits that the § 103(a) rejection is improper and should be reversed.

B. The § 103(A) Rejection Of Claims 8-10 And 17-18 Over Mitsuhiro In View Of Standard Register Use (As Applied Above), And In Further View Of Fujimura Is Improper And Should Be Reversed.

Appellant respectfully submits that the § 103(a) rejection of claims 8-10 and 17-18 is improper for at least the reason that it is based on the improper combination of Mitsuhiro and Yoshida, discussed in Section A above, and that the Fujimura reference appears to provide no teaching or disclosure to cure the noted underlying deficiencies of Mitsuhiro and Yoshida.

Fujimura is cited for its alleged disclosure of returning from an interrupt and restoring existing conditions, which the Examiner admits the underlying references fail to teach. However, the Examiner has made no effort to analyze how the elements purportedly taught by Fujimura are to be combined with Mitsuhiro and Yoshida. For example, it is unclear how the combination of references would correspond to the feature recited in claim 8 directed to register bank block selection data indicative of a pre interrupt switch state. As another example, neither the Examiner (nor the applied references) clearly indicate what would correspond to a pre interrupt register bank block selection signal derived from the stored register bank block selection data, as recited in claim 8. Appellant submits that the Examiner has failed to provide support for: 1) what aspects of the Fujimura reference correspond to the various claim recitations; and 2) how these aspects would function in the asserted circuit of the Mitsuhiro reference. Without a showing of correspondence for each claim limitation and an analysis of the limitations as a whole, Appellant submits that the rejections cannot stand.

For at least these reasons, Appellant submits that the § 103(a) rejection of claims 8-10 and 17-18 is improper and should be reversed.

C. The § 103(A) Rejection Of Claim 12 Over Mitsuhira In View Of Standard Register Use (As Applied Above), And In Further View Of Hohl Is Improper And Should Be Reversed.

Appellant respectfully submits that the § 103(a) rejection of claim 12 is improper for at least the reason that it is based on the improper combination of Mitsuhira and Yoshida, discussed in Section A above, and that the Hohl reference appears to provide no teaching or disclosure to cure the noted underlying deficiencies of Mitsuhira and Yoshida.

Hohl is cited for its alleged disclosure of accessing register data during debugging, which the Examiner admits the underlying references fail to teach. However, as with the other § 103(a) rejections discussed above, the Examiner has made no effort to analyze how the elements purportedly taught by Hohl are to be combined with Mitsuhira and Yoshida in a way that would result in the claimed invention. Appellant submits that the Examiner has failed to provide support for: 1) what aspects of the Hohl reference correspond to the various claim recitations; and 2) how these aspects would function in the asserted circuit of the Mitsuhira reference. Without a showing of correspondence for each claim limitation and an analysis of the limitations as a whole, Appellant submits that the rejections cannot stand.

For at least these reasons, Appellant submits that the § 103(a) rejection of claim 12 is improper and should be reversed.

VIII. Conclusion

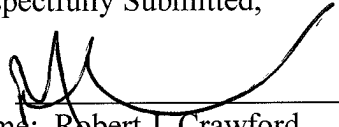
In view of the above, Appellant submits that the rejections of claims 1-21 are improper. Appellant therefore requests reversal of the rejections as applied to the appealed claims and allowance of the entire application. Authority to charge the undersigned's deposit account was provided on the first page of this brief.

Please direct all correspondence to:

Corporate Patent Counsel
NXP Intellectual Property & Standards
1109 McKay Drive; Mail Stop SJ41
San Jose, CA 95131

CUSTOMER NO. 65913

Respectfully Submitted,

By: 
Name: Robert J. Crawford
Reg. No.: 32,122
Tel: 651 686-6633 ext. 2300
(NXPS.281PA)

APPENDIX OF CLAIMS INVOLVED IN THE APPEAL
(S/N 10/566,515)

1. A system comprising:
 - a processor that executes logical or arithmetic operations;
 - a plurality of register bank blocks used as special function registers by the processor during the execution of the logical or arithmetic operations and,
 - a register bank block decoder circuit for activating one and only one of the plurality of register bank blocks, the register bank block decoder circuit responsive to interrupt event operations for selecting the one of the plurality of register bank blocks for being activated, where different interrupt event operations result in selection of different ones of the plurality of register bank blocks.
2. A system according to claim 1, further comprising:
 - a memory circuit for storing of a first program stream and for storing of a second program stream, wherein the processor utilizes a first register bank block from the plurality of register bank blocks during execution of the first program stream, and upon the occurrence of an interrupt resulting from an interrupt event associated with the second program stream, the processor executes the second program stream utilizing a second register bank block, the second register bank block different and logically isolated from the first register bank block.
3. A system according to claim 2, wherein the second program stream has a higher interrupt priority than the first program stream.
4. A system according to claim 1, further comprising:
 - an input data bus and,
 - an input switching circuit coupled to the plurality of register bank blocks and having a selection input port for receiving a register bank block selection signal from the register bank block decoder circuit, the input switching circuit for activating one of the plurality of register

bank blocks in dependence upon the register bank block selection signal, the activated one of the plurality of register bank blocks being coupled to the input data bus.

5. A system according to claim 4, wherein the input switching circuit is a multiplexer circuit.
6. A system according to claim 4, further comprising:
an output data bus; and,
an output switching circuit coupled to the plurality of register bank blocks and having a selection input port for receiving the register bank block selection signal from the register bank block decoder circuit, the output switching circuit for switchably coupling the activated one of the plurality of register bank blocks to the output data bus.
7. A system according to claim 6, wherein the output switching circuit is a multiplexer circuit.
8. A system according to claim 6, further comprising a circuit for storing and retrieving of register bank block selection data derived from the register bank block selection signal, the register bank block selection data indicative of a pre interrupt switch state, wherein upon terminating of an interrupt event, the input switching circuit and the output switching circuit are provided with a pre interrupt register bank block selection signal derived from the stored register bank block selection data.
9. A system according to claim 8, wherein the state of the circuit for storing and retrieving of the register bank block selection data is based on interrupt priority.
10. A system according to claim 6, wherein the register bank block selection signal is based solely on interrupt priority.

11. A system according to claim 1, wherein a first register bank block from the plurality of register bank blocks is concurrently enabled along with a second different register bank block from the plurality of register bank blocks, the second different register bank block independently addressable from the first register bank block.
12. A system according to claim 1, further comprising a debug bank select register coupled to the register bank block decoder circuit, the debug bank select register for providing access to program stream data stored within the plurality of register bank blocks during a step of debugging.
13. A method of switching processing resources in a data processing system comprising the steps of:
 - providing a plurality of register bank blocks;
 - utilizing a first register bank block from the plurality of register bank blocks as special function registers during execution of logical or arithmetic operations;
 - receiving of an interrupt request for initiating an interrupt event;
 - determining if the interrupt request is to be fulfilled, and if so, then: selecting a second register bank block from the plurality of register bank blocks, the selected second register bank block in isolation from the first register bank block; and,
 - utilizing the second register bank block from the plurality of register bank blocks as special function registers during execution of logical or arithmetic operations.
14. A method according to claim 13, wherein a first program stream is provided for utilizing of the first register bank block and a second program stream is provided for utilizing the second register bank block.
15. A method according to claim 14, wherein the first program stream has a lower interrupt priority than the second program stream, the interrupt priority used in the step of determining whether to fulfill the interrupt request.

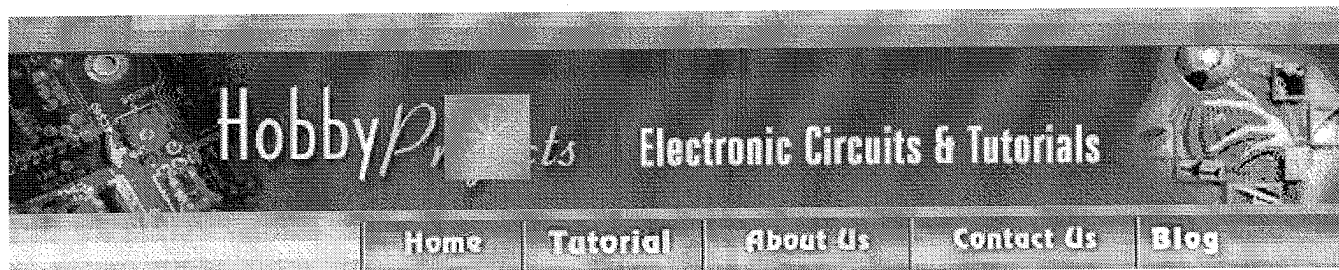
16. A method according to claim 14, wherein a processor executes the first and second program streams.
17. A method according to claim 16, further comprising the steps of:
halting execution of the second program stream;
selecting the first register bank block; and,
resuming execution of the first program stream.
18. A method according to claim 17, wherein executing the instructions of the second program stream takes place without altering the contents of the first register bank block in suspended use by the first program stream.
19. A method according to claim 14, further comprising the step of:
providing a memory circuit having a first memory region for storing of program stream data related to the first program stream.
20. A method according to claim 14, wherein the first and second program streams have stored therein instruction data for storing and restoring of register bank block contents.
21. A storage medium having data stored thereon, the data for implementation of a processing system comprising:
first instruction data for providing a plurality of register bank blocks that are used as special function registers during execution of logical or arithmetic operations and,
second instruction data for providing a register bank block decoder circuit for activating one of the plurality of register bank blocks in isolation, the register bank block decoder circuit responsive to interrupt event operations for selecting the one of the plurality of register bank blocks for being activated, where different interrupt event operations result in selection of different ones of the plurality of register bank blocks.

APPENDIX OF EVIDENCE

Appellant is unaware of any evidence submitted in this application pursuant to 37 C.F.R. §§ 1.130, 1.131, and 1.132.

APPENDIX OF RELATED PROCEEDINGS

As stated in Section II above, Appellant is unaware of any related appeals, interferences or judicial proceedings.



Electronic, Microprocessor, Micro Controller and PC based Projects / Circuits for Engineering Students, Hobbyists and F

MICROCONTROLLER TUTORIALS - 8051

Tutorials

Google

Search

☐ Web ☒ Hobbyprojects.com

8051 - Special Function Registers

What Are SFRs?

The 8051 is a flexible microcontroller with a relatively large number of modes of operations. Your program may inspect and/or change the operating mode of the 8051 by manipulating the values of the 8051's Special Function Registers (SFRs).

SFRs are accessed as if they were normal Internal RAM. The only difference is that Internal RAM is from address 00h through 7Fh whereas SFR registers exist in the address range of 80h through FFh.

Each SFR has an address (80h through FFh) and a name. The following chart provides a graphical presentation of the 8051's SFRs, their names, and their address.

80	PC	SC	DPL	DPH				PCON	87
88	ICON	IMOD	TL0	TL1	TH0	TH1			8F
90	P1								97
98	SCON	SBUP							9F
A0	P2								A7
A8	IP								AF
B0	P3								B7
B8	IP								B9
C0									C7
C8									CF
D0	PSW								D7
D8									DF
E0	ACC								E7
E8									EF
F0									F7
F8									FF

Blue background are I/O port SFRs
 Yellow background are control SFRs
 Green background are other SFRs

Free 8051 Resource

Info on hardware, software, tools, services and valuable articles
www.EECatalog.com

8051

Microcontroller

Revolutionary Design & Performance. Cut Design Time While Saving Costs.
www.Cypress.com

8051 development tools

Compiler, Assembler, Simulator, IDE Free 4kb version. CodeCompressor.
www.8051tools.com

DB9 / DB25 Serial Cables

Our serial cables UL listed, lifetime warranty.
www.cablewholesale.com

Circuits

8051

Introduction

Chapter 1 Types of Memory

Chapter 2
Special
Function
Registers

Chapter 3
Basic
Registers

Chapter 4
Addressing
Modes

Chapter 5
Program Flow

Chapter 6
Low Level
Information

Chapter 7
Timers

Chapter 8
Serial Port
Operations

Chapter 9
Interrupts

Additional
Features in
8052

8052
Instruction
Set

As you can see, although the address range of 80h through FFh offer 128 possible addresses, there are only 21 SFRs in a standard 8051. All other addresses in the SFR range (80h through FFh) are considered invalid. Writing to or reading from these registers may produce undefined values or behavior.

Programming Tip: It is recommended that you not read or write to SFR addresses that have not been assigned to an SFR. Doing so may provoke undefined behavior and may cause your program to be incompatible with other 8051-derivatives that use the given SFR for some other purpose.

SFR Types

As mentioned in the chart itself, the SFRs that have a blue background are SFRs related to the I/O ports. The 8051 has four I/O ports of 8 bits, for a total of 32 I/O lines. Whether a given I/O line is high or low and the value read from the line are controlled by the SFRs in green.

The SFRs with yellow backgrounds are SFRs which in some way control the operation or the configuration of some aspect of the 8051. For example, **TCON** controls the timers, **SCON** controls the serial port.

The remaining SFRs, with green backgrounds, are "other SFRs." These SFRs can be thought of as auxiliary SFRs in the sense that they don't directly configure the 8051 but obviously the 8051 cannot operate without them. For example, once the serial port has been configured using **SCON**, the program may read or write to the serial port using the **SBUF** register.

Programming Tip: The SFRs whose names appear in red in the chart above are SFRs that may be accessed via bit operations (i.e., using the **SETB** and **CLR** instructions). The other SFRs cannot be accessed using bit operations. As you can see, all SFRs that whose addresses are divisible by 8 can be accessed with bit operations.

SFR Descriptions

This section will endeavor to quickly overview each of the standard SFRs found in the above SFR chart map. It is not the intention of this section to fully explain the functionality of each SFR--this information will be covered in separate chapters of the tutorial. This section is to just give you a general idea of what each SFR does.

P0 (Port 0, Address 80h, Bit-Addressable): This is input/output port 0. Each bit of this SFR corresponds to one of the pins on the microcontroller. For example, bit 0 of port 0 is pin P0.0, bit 7 is pin P0.7. Writing a value of 1 to a bit of this SFR will send a high level on the corresponding I/O pin whereas a value of 0 will bring it to a low level.

Programming Tip: While the 8051 has four I/O port (P0, P1, P2, and P3), if your hardware uses external RAM or external code memory (i.e., your program is stored in an external ROM or EPROM chip or if you are using external RAM chips) you may not use P0 or P2. This is because the 8051 uses ports P0 and P2 to address the external memory. Thus if you are using external RAM or code memory you may only use ports P1 and P3 for your own use.

SP (Stack Pointer, Address 81h): This is the stack pointer of the microcontroller. This SFR indicates where the next value to be taken from the stack will be read from in Internal RAM. If you push a value onto the stack, the value will be written to the address of SP + 1. That is to say, if SP holds the value

07h, a PUSH instruction will push the value onto the stack at address 08h. This SFR is modified by all instructions which modify the stack, such as PUSH, POP, LCALL, RET, RETI, and whenever interrupts are provoked by the microcontroller.

Programming Tip: The SP SFR, on startup, is initialized to 07h. This means the stack will start at 08h and start expanding upward in internal RAM. Since alternate register banks 1, 2, and 3 as well as the user bit variables occupy internal RAM from addresses 08h through 2Fh, it is necessary to initialize SP in your program to some other value if you will be using the alternate register banks and/or bit memory. It's not a bad idea to initialize SP to 2Fh as the first instruction of every one of your programs unless you are 100% sure you will not be using the register banks and bit variables.

DPL/DPH (Data Pointer Low/High, Addresses 82h/83h): The SFRs DPL and DPH work together to represent a 16-bit value called the *Data Pointer*. The data pointer is used in operations regarding external RAM and some instructions involving code memory. Since it is an unsigned two-byte integer value, it can represent values from 0000h to FFFFh (0 through 65,535 decimal).

Programming Tip: DPTR is really DPH and DPL taken together as a 16-bit value. In reality, you almost always have to deal with DPTR one byte at a time. For example, to push DPTR onto the stack you must first push DPL and then DPH. You can't simply push DPTR onto the stack. Additionally, there is an instruction to "increment DPTR." When you execute this instruction, the two bytes are operated upon as a 16-bit value. However, there is no instruction that decrements DPTR. If you wish to decrement the value of DPTR, you must write your own code to do so.

PCON (Power Control, Addresses 87h): The Power Control SFR is used to control the 8051's power control modes. Certain operation modes of the 8051 allow the 8051 to go into a type of "sleep" mode which requires much less power. These modes of operation are controlled through PCON. Additionally, one of the bits in PCON is used to double the effective baud rate of the 8051's serial port.

TCON (Timer Control, Addresses 88h, Bit-Addressable): The Timer Control SFR is used to configure and modify the way in which the 8051's two timers operate. This SFR controls whether each of the two timers is running or stopped and contains a flag to indicate that each timer has overflowed. Additionally, some non-timer related bits are located in the TCON SFR. These bits are used to configure the way in which the external interrupts are activated and also contain the external interrupt flags which are set when an external interrupt has occurred.

TMOD (Timer Mode, Addresses 89h): The Timer Mode SFR is used to configure the mode of operation of each of the two timers. Using this SFR your program may configure each timer to be a 16-bit timer, an 8-bit autoreload timer, a 13-bit timer, or two separate timers. Additionally, you may configure the timers to only count when an external pin is activated or to count "events" that are indicated on an external pin.

TLO/TH0 (Timer 0 Low/High, Addresses 8Ah/8Ch): These two SFRs, taken together, represent timer 0. Their exact behavior depends on how the timer is configured in the TMOD SFR; however, these timers always count up. What is configurable is how and when they increment in value.

TL1/TH1 (Timer 1 Low/High, Addresses 8Bh/8Dh): These two SFRs, taken together, represent timer 1. Their exact behavior depends on how the timer is configured in the TMOD SFR; however, these timers always count up. What is

configurable is how and when they increment in value.

P1 (Port 1, Address 90h, Bit-Addressable): This is input/output port 1. Each bit of this SFR corresponds to one of the pins on the microcontroller. For example, bit 0 of port 1 is pin P1.0, bit 7 is pin P1.7. Writing a value of 1 to a bit of this SFR will send a high level on the corresponding I/O pin whereas a value of 0 will bring it to a low level.

SCON (Serial Control, Addresses 98h, Bit-Addressable): The Serial Control SFR is used to configure the behavior of the 8051's on-board serial port. This SFR controls the baud rate of the serial port, whether the serial port is activated to receive data, and also contains flags that are set when a byte is successfully sent or received.

Programming Tip: To use the 8051's on-board serial port, it is generally necessary to initialize the following SFRs: SCON, TCON, and TMOD. This is because SCON controls the serial port. However, in most cases the program will wish to use one of the timers to establish the serial port's baud rate. In this case, it is necessary to configure timer 1 by initializing TCON and TMOD.

SBUF (Serial Control, Addresses 99h): The Serial Buffer SFR is used to send and receive data via the on-board serial port. Any value written to SBUF will be sent out the serial port's TXD pin. Likewise, any value which the 8051 receives via the serial port's RXD pin will be delivered to the user program via SBUF. In other words, SBUF serves as the output port when written to and as an input port when read from.

P2 (Port 2, Address A0h, Bit-Addressable): This is input/output port 2. Each bit of this SFR corresponds to one of the pins on the microcontroller. For example, bit 0 of port 2 is pin P2.0, bit 7 is pin P2.7. Writing a value of 1 to a bit of this SFR will send a high level on the corresponding I/O pin whereas a value of 0 will bring it to a low level.

Programming Tip: While the 8051 has four I/O ports (P0, P1, P2, and P3), if your hardware uses external RAM or external code memory (i.e., your program is stored in an external ROM or EPROM chip or if you are using external RAM chips) you may not use P0 or P2. This is because the 8051 uses ports P0 and P2 to address the external memory. Thus if you are using external RAM or code memory you may only use ports P1 and P3 for your own use.

IE (Interrupt Enable, Addresses A8h): The Interrupt Enable SFR is used to enable and disable specific interrupts. The low 7 bits of the SFR are used to enable/disable the specific interrupts, whereas the highest bit is used to enable or disable ALL interrupts. Thus, if the high bit of IE is 0 all interrupts are disabled regardless of whether an individual interrupt is enabled by setting a lower bit.

P3 (Port 3, Address B0h, Bit-Addressable): This is input/output port 3. Each bit of this SFR corresponds to one of the pins on the microcontroller. For example, bit 0 of port 3 is pin P3.0, bit 7 is pin P3.7. Writing a value of 1 to a bit of this SFR will send a high level on the corresponding I/O pin whereas a value of 0 will bring it to a low level.

IP (Interrupt Priority, Addresses B8h, Bit-Addressable): The Interrupt Priority SFR is used to specify the relative priority of each interrupt. On the 8051, an interrupt may either be of low (0) priority or high (1) priority. An interrupt may only interrupt interrupts of lower priority. For example, if we configure the 8051

so that all interrupts are of low priority except the serial interrupt, the serial interrupt will always be able to interrupt the system, even if another interrupt is currently executing. However, if a serial interrupt is executing no other interrupt will be able to interrupt the serial interrupt routine since the serial interrupt routine has the highest priority.

PSW (Program Status Word, Addresses D0h, Bit-Addressable): The Program Status Word is used to store a number of important bits that are set and cleared by 8051 instructions. The PSW SFR contains the carry flag, the auxiliary carry flag, the overflow flag, and the parity flag. Additionally, the PSW register contains the register bank select flags which are used to select which of the "R" register banks are currently selected.

Programming Tip: If you write an interrupt handler routine, it is a very good idea to *always* save the PSW SFR on the stack and restore it when your interrupt is complete. Many 8051 instructions modify the bits of PSW. If your interrupt routine does not guarantee that PSW is the same upon exit as it was upon entry, your program is bound to behave rather erratically and unpredictably--and it will be tricky to debug since the behavior will tend not to make any sense.

ACC (Accumulator, Addresses E0h, Bit-Addressable): The Accumulator is one of the most-used SFRs on the 8051 since it is involved in so many instructions. The Accumulator resides as an SFR at E0h, which means the instruction **MOV A,#20h** is really the same as **MOV E0h,#20h**. However, it is a good idea to use the first method since it only requires two bytes whereas the second option requires three bytes.

B (B Register, Addresses F0h, Bit-Addressable): The "B" register is used in two instructions: the multiply and divide operations. The B register is also commonly used by programmers as an auxiliary register to temporarily store values.

Other SFRs

The chart above is a summary of all the SFRs that exist in a standard 8051. All derivative microcontrollers of the 8051 must support these basic SFRs in order to maintain compatibility with the underlying MCS51 standard.

A common practice when semiconductor firms wish to develop a new 8051 derivative is to add additional SFRs to support new functions that exist in the new chip.

For example, the Dallas Semiconductor DS80C320 is upwards compatible with the 8051. This means that any program that runs on a standard 8051 should run without modification on the DS80C320. This means that all the SFRs defined above also apply to the Dallas component.

However, since the DS80C320 provides many new features that the standard 8051 does not, there must be some way to control and configure these new features. This is accomplished by adding additional SFRs to those listed here. For example, since the DS80C320 supports two serial ports (as opposed to just one on the 8051), the SFRs SBUF2 and SCON2 have been added. In addition to all the SFRs listed above, the DS80C320 also recognizes these two new SFRs as valid and uses their values to determine the mode of operation of the secondary serial port. Obviously, these new SFRs have been assigned to SFR addresses that were unused in the original 8051. In this manner, new 8051 derivative chips may be developed which will run existing 8051 programs.

Programming Tip: If you write a program that utilizes new SFRs that are specific to a given derivative chip and not included in the above SFR list, your program will not run properly on a standard 8051 where that SFR does not exist. Thus, only use non-standard SFRs if you are sure that your program will only have to run on that specific microcontroller. Likewise, if you write code that uses non-standard SFRs and subsequently share it with a third-party, be sure to let that party know that your code is using non-standard SFRs to save them the headache of realizing that due to strange behavior at run-time.

<<< [Click here to come back on \(Special Function Registers\)](#)

<<<< [Back to 8051 Microcontroller Tutorial](#)

